

Second Hand Market

REPORT



Institute of Engineering & Technology

Team Members

Muskan Upadhyay
(181500407)

Nitin Kumar Singh
(181500434)

Contents

Abstract

1. Introduction

1.1 General Introduction to the topic

1.2 Introduction of technology

1.2.1 Introduction of MEAN stack

1.2.2 Division of MEAN

1.3 Hardware and Software Requirements

1.4 Modules & their functionalities

2. Software Design

2.1 Use Case Diagram

2.2 Data Flow Diagram

2.2.1 Level-0 Diagram

2.2.2 Level-1 Diagram

2.3 Entity Relationship Diagram

3. Creating a MEAN Project

4. Coding the Web

5. Testing & Snapshots

6. Difficulties Encountered**7. Contribution of Group Members****8. Future Scope****9. References**

Abstract

Proposed Second Hand Market is a platform for buying and selling services and goods such as electronics, fashion items, furniture, household goods, cars and bikes. It is a web portal that provides an easy way to sell and buy old products. If you have any product that you would like to sell, you can list them in the Sell Product section. Alternately, if you wish to buy, you can search the Buy Product section. SHM operates an online marketplace for consumer-to-consumer sales, particularly targeting users in emerging markets and it is not going to handle any payment processing, with a view to providing a safe, reliable and efficient way for consumers to buy and sell goods. We can't trust everyone that the owner of the product will deliver the product after getting money of it, so we simply tie up with courier service for collecting products, delivering products and collecting money from the buyer. After selecting the product for buying and its confirmation, courier person will take care of movement of products from seller to buyer.

In present scenario, we are dealing with many frauds and hindrances while using websites like Olx, Quikr etc. when it comes to safe and reliable payment processing. It may happen at times that the owner of the product will not deliver the product after getting money. And sometimes the receiver takes the product and then refuse to give money. Over the years, OLX and Quikr has become a common place for fraud, especially sellers who post fake advertisements to dupe buyers upon receiving advance payment and

fraudulent buyers who engage in UPI scam, phishing and sending fake SMSes and emails of bank transfer confirmation.

So, to handle this issue we tie up with courier service who will take cash on delivery from the buyer after taking booking from seller hence no need to make any online payment.

Introduction

1.1 General Introduction to the topic

All people want to sell their old products but they don't find a suitable customer to sell their product. Also, there are people who can't afford to buy new products at high prices. This website is designed to meet the requirements of both types of people making it easier for them to sell and buy their products at suitable price without any extra charges.

- ✓ Buy products at affordable rates.
- ✓ Sell at rates of your own choice.
- ✓ No wait to see the products if someone else are taking that.
- ✓

Therefore, I came up with this project.

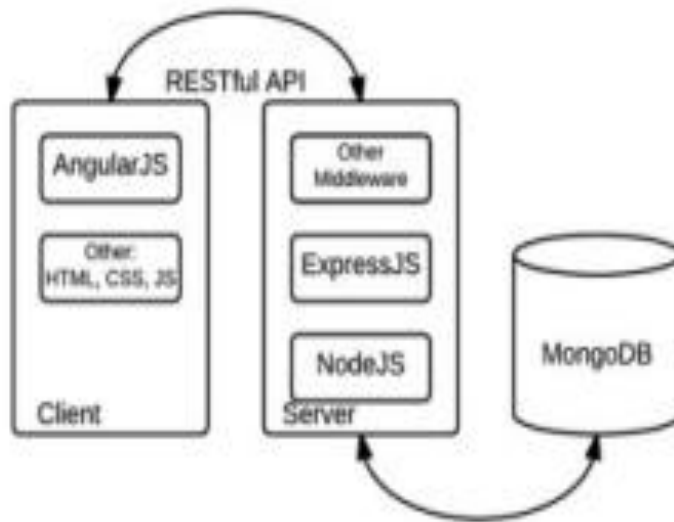
- The SHM will be design to provide a complete system for controlling and monitoring buying and selling products online.
- Buying and selling of products module is used to maintain member registration, product search, listing of products to sell, buy product, display order status.
- All products are going to be categorized for easy search, in home page's latest tab and top 10 products will be displayed in it.

- The buying and selling of products would be done without any online payment processing. For this, we used payment on delivery concept where buyer gives the payment after taking the book from delivery person.

1.2 Introduction of Technology:

1.2.1 Introduction of MEAN Stack

MEAN is used to build modern web applications. In this blog, we are going to get a brief introduction on MEAN and how to install it in your system. As the acronym of MEAN is MongoDB, Node.js, Express, and Angular, it includes all the 4 technologies combined into it. It is designed to give a quick and organized way to develop MEAN based web apps, websites, web services and APIs. Using MEAN, developers can set up their work with a set of popular tools, which were carefully combined, so the developers need not concentrate on never ending work of system administration, package management, libraries, etc. Instead, they can concentrate on development completely.



Now let's look into individual technologies included in MEAN.

1.2.2 Division of MEAN

MongoDB

MEAN is a collection of MongoDB, Express, Angular, Node.js. It includes all the 4 technologies combined into it. Let's see each in brief. MongoDB MongoDB is an open source, NoSQL database designed for cloud applications. It uses object-oriented organization instead of a relational model. In the MEAN stack, MongoDB stores the application's data. Because both the application and the database use JavaScript, there's no need to translate the object as it journeys from the application to the database and back. The application can push and pull objects between the back end and the database without missing a beat.

MongoDB is known for its scalability in both storage and performance. You can add fields to the database without reloading the entire table, and MongoDB is well known for its ability to manage large amounts of data without compromising on data access. With just a few clicks, you can expand the resources available to your database, making it perfect for applications with occasional periods of increased activity.

MongoDB's document model is simple for developers to learn and use, while still providing all the capabilities needed to meet the most complex requirements at any scale. We provide drivers for 10+ languages, and the community has built dozens more.

MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time. The document model maps to the objects in your application code, making data easy to work with. Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyse your data. MongoDB is a distributed database at its core, so high availability, horizontal scaling, and geographic distribution are built in and easy to use. MongoDB is free to use. Versions released prior to October 16, 2018 are published under the AGPL.

Express

Express is one the most popular and widely used web frameworks in Node.js development zone. Express is a minimal web server built on Node.js that provides all the essential functionality required for delivering web applications to the browser and mobile devices. Express.js allows you to handle Routes, Server, and I/O stuff very easily.

Angular 8.0

Angular used to be the “golden child” among JavaScript frameworks, as it was initially introduced by Google corporation. It was built with the Model-View-Controller concept in mind, though authors of the framework often called it “Model-View. The framework, written in pure JavaScript, was intended to decouple an application’s logic from DOM manipulation, and aimed at dynamic page updates. Still, it wasn’t very intrusive: you could have only a part of the page controlled by Angular. This framework introduced many powerful features allowing the developer to create rich, single-page applications quite easily. Specifically, an interesting concept of data binding was introduced that meant automatic updates of the view whenever the model (data) changed, and vice versa.

Angular new project steps:

1. `npm install -g angular-cli`
2. `ng new my_first_angular_app`
3. `ng new my_first_angular_app --style=scss`
4. `cd my_first_angular_app`
5. `ng serve`

Node.js

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

Used Node packages:

1. **Cors:** CORS is a node.js package for providing a **Connect/Express** middleware.
2. **Body-Parser:** Parse incoming request bodies in a middleware before your handlers, available under the req.body property.
3. **Nodemailer:** Nodemailer package is used to sending emails.
4. **Multer:** Multer is used to upload the images to the databases.
5. **Sha1:** Sha1 is used to encrypt the password fields before save in database.

Other Used Languages:

1.**HTML:** HTML stands for Hyper Text Markup Language. It is used to design web pages using Markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages.

2. CSS: Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

3. Bootstrap 4: Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first web sites. It solves many problems which we had once, one of which is the cross-browser compatibility issue.

4. JavaScript: JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform. JavaScript is the most popular programming language in the world and that makes it a programmer's great choice. Once you learnt JavaScript, it helps you developing great front-end as well as back-end softwares using different JavaScript based frameworks like jQuery, Node.JS etc.

1.3 Hardware & Software Requirements:

Software Specifications –

- **Technology Implemented:**

1. **MongoDB** - MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need.
2. **Express.js** – Express.js is a Node.js web application framework, which is specifically designed for building single-page, multi-page and hybrid web applications.
3. **Angular 7** – Angular 7 is a frontend framework.
4. **Node.js** – we are using Node.js as backend technology. It is an open source server environment. It uses java script on server.

The combination of these four technologies is called ‘**MEAN Stack**’.

HTML: For user interface.

CSS: For making interfaces more attractive and stylish.

Bootstrap 4: For making website responsive and more attractive.

- **Language Used:** JavaScript
- **Database:** MongoDB
- **User Interface Design:** Website
- **Web Browser:** Chrome

HARDWARE REQUIREMENTS -

- **Processor:** intel i5
- **Operating System:** Windows 10
- **RAM:** 8GB
- **Hardware Devices:** Computer System
- **Hard Disk:** 256 GB

1.4 Modules and their functionalities

Admin Panel:

1. Admin Login: Using this module, admin can login to the admin panel.
2. Admin Main: After logging in, the admin will be in the main section of the admin panel.
3. Change Password: In case admin forgets his current password or he wants to change his current password, he can change using this module.
4. Approve Member: Admin will approve all the new users who will sign up their account on the website.
5. Approve Product: Before uploading any old product on the website, admin will first approve the product.

Buyer and Seller:

1. Default Home Page: The page that will be opened at first when we open the website whether a person has an existing account on the website or not.
2. Member Login: Any member who wants to buy or sell an old item, first will login to the website using this module.

3. Forget and Change Password: Any user having an existing account can change his password in case he forgets his current password or if he just want to change for security purpose.
4. Chat between buyer and seller: If one likes a product, he can negotiate with the seller of the product in the chat section.
5. Edit Profile: The person with an existing account on the website can update his profile like name, address or phone number anytime whenever he wants.

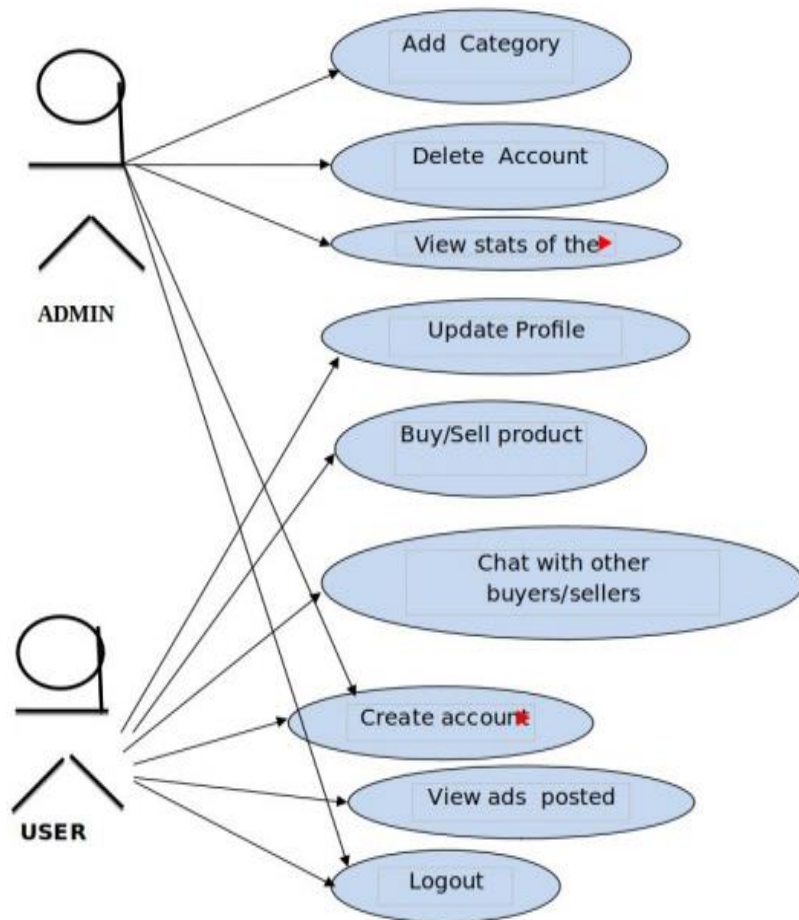
Administrator:

Administrator can login through the admin panel. The administrator will be able to add categories. He/she will be able to see the new registered users. The administrator can also see the feedback given by the buyers.

Software Design

4.1 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

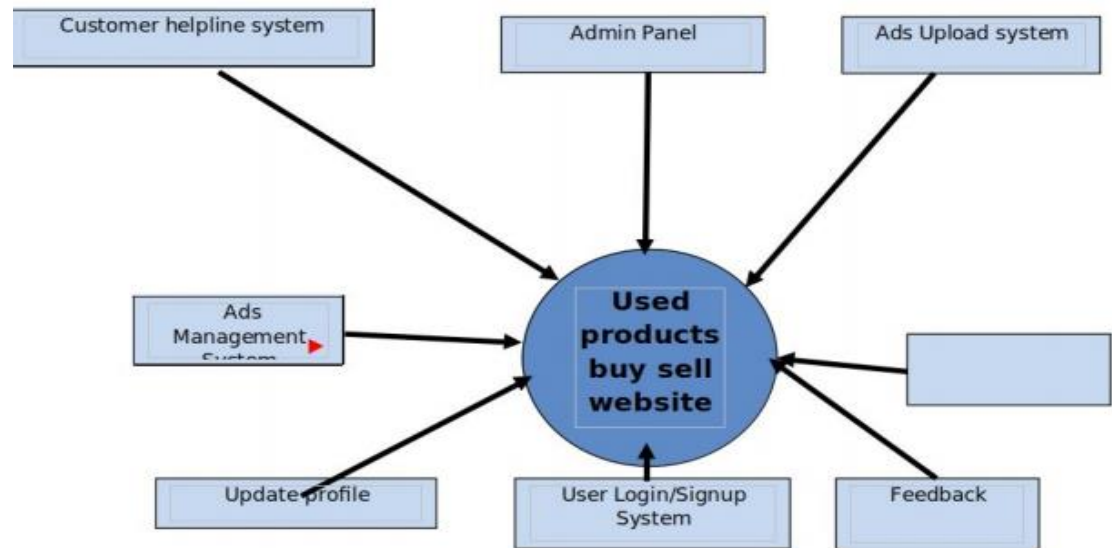


4.2 Data Flow Diagram

Data Flow diagrams show the flow of data from external entities into the system, and from one process to another within the system.

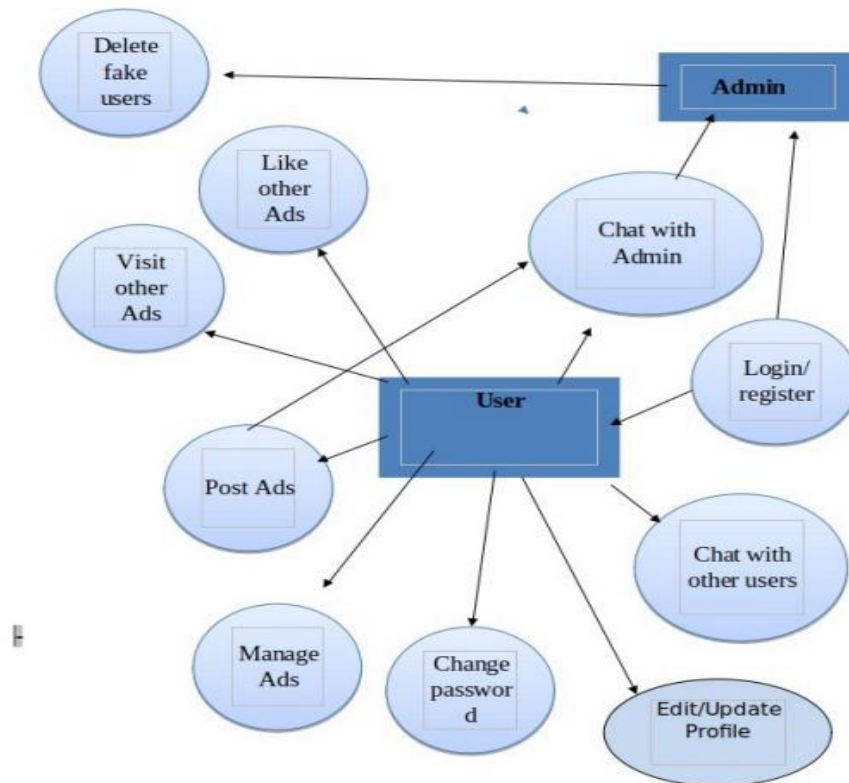
4.2.1 Level -0 Diagram:

The level 0 diagram provides a conceptual view of the process and its surrounding input, output and data stores. It is called context level Data flow diagram also.

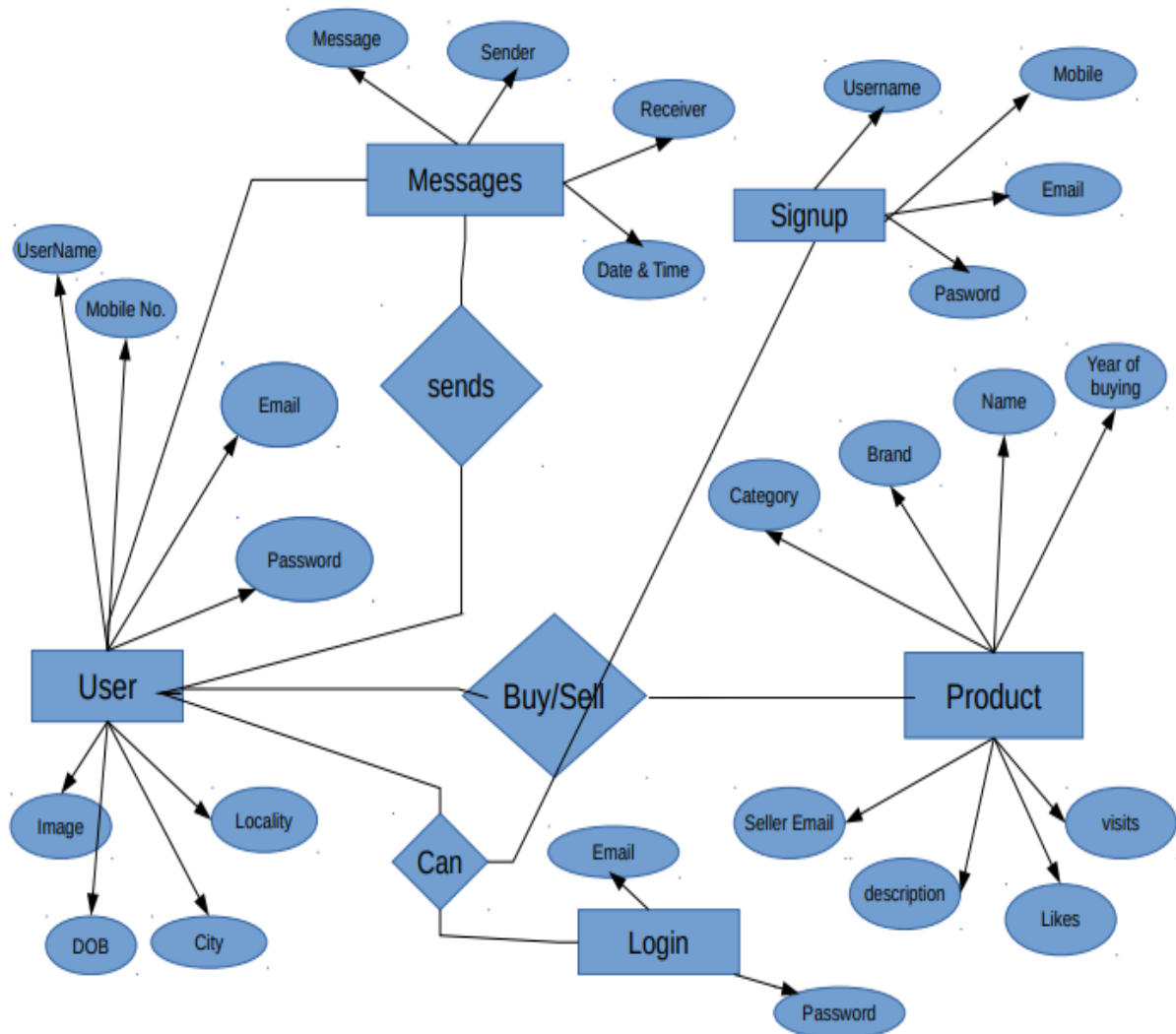


4.2.2 Level -1 Diagram:

The level -1 diagram provides a more detailed and comprehensive view of the interaction among the sub-processes within the system.

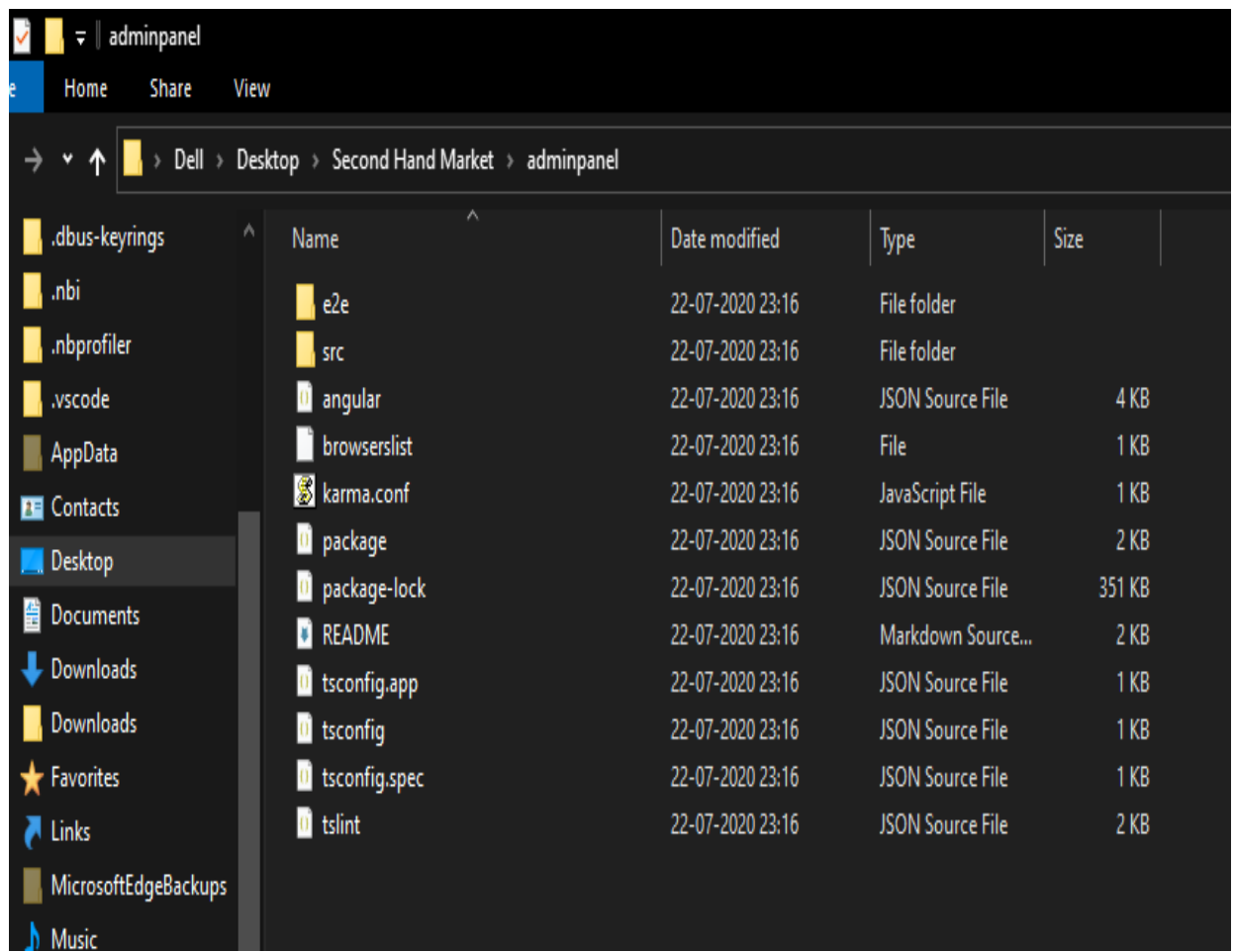


4.3 Entity Relationship Diagram



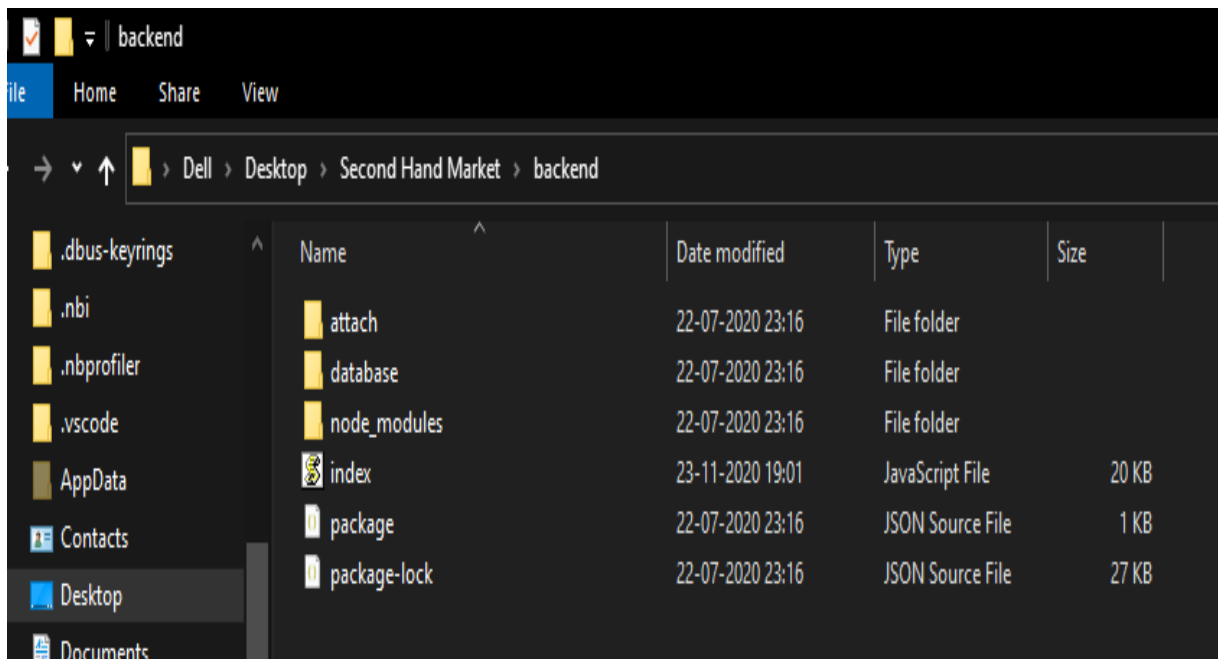
Creating a MEAN Project

- Create a separate folder named **Second Hand Market** to store all the project files.
- Go inside this folder open cmd and run the command
`ng new Frontend`
- The above command will create a new **Angular** project named **Frontend** containing all the files shown below.



****Important:** You should have installed latest version of node and npm on your system.

- Run the same command again to generate another Angular project to design the AdminPanel.
- Now create another folder inside Second Hand Market folder named Backend to store all the backend data of the site.
- Go inside the Backend folder and open cmd and run the command
npm init
- This will generate a package.json file, open the package.json file and set all the dependencies.
- After this run the command to install all the dependencies npm install –g node-modules.
- Now your backend folder will load like as shown below.



Coding the Web

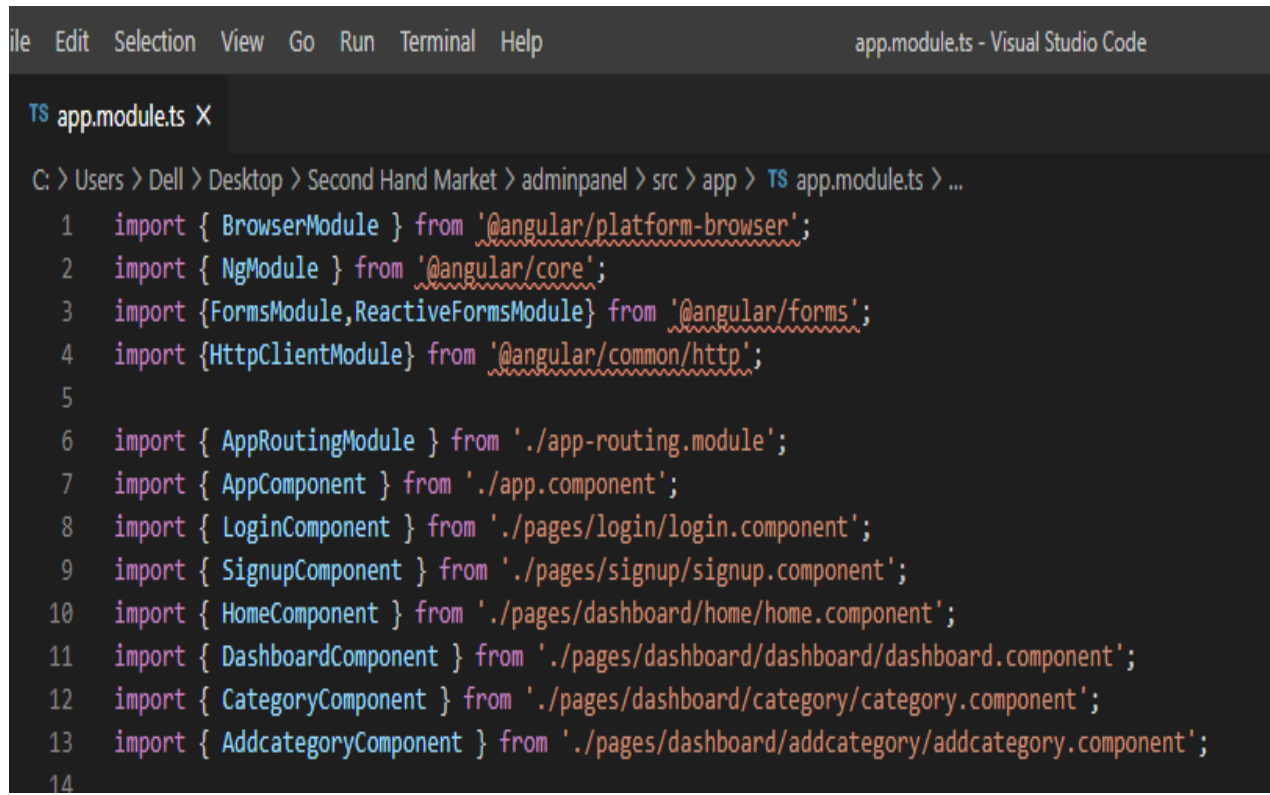
Let's setup Adminpanel

- Go inside the AdminPanel folder and open the folder in cmd and run the command `ng serve`. This will compile your code.
- Go to Chrome browser (or any other) and type `localhost:4200` and you will see something like this as shown below:



- Now create a Login component to allow the Admin to login to admin panel.

- To create a component open adminpanel folder in cmd and type command `ng g c login`, this will automatically create the component and add it to the `app.module.ts` file as show in image below.



```
TS app.module.ts X
C: > Users > Dell > Desktop > Second Hand Market > adminpanel > src > app > TS app.module.ts > ...
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { FormsModule, ReactiveFormsModule } from '@angular/forms';
4  import { HttpClientModule } from '@angular/common/http';
5
6  import { AppRoutingModule } from './app-routing.module';
7  import { AppComponent } from './app.component';
8  import { LoginComponent } from './pages/login/login.component';
9  import { SignupComponent } from './pages/signup/signup.component';
10 import { HomeComponent } from './pages/dashboard/home/home.component';
11 import { DashboardComponent } from './pages/dashboard/dashboard/dashboard.component';
12 import { CategoryComponent } from './pages/dashboard/category/category.component';
13 import { AddcategoryComponent } from './pages/dashboard/addcategory/addcategory.component';
14
```

`approuting.module.ts`

This file is used to store all the routes. After creating all the components needed you can give routes to them.

Adding Bootstrap, Fonts and other Styles

You can add bootstrap, fonts, Styles etc to Adminpanel/src/app/index.html file as shown below:


```

1  <!doctype html>
2  <html lang="en">
3  <head>
4    <meta charset="utf-8">
5    <title>Adminpanel</title>
6    <base href="/">
7
8    <meta name="viewport" content="width=device-width, initial-scale=1">
9    <link rel="icon" type="image/x-icon" href="favicon.ico">
10   <link rel="stylesheet" href="assets/css/bootstrap.css">
11   <link href="https://fonts.googleapis.com/css?family=Jura&display=swap" rel="stylesheet">
12   <link href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet" integrity="sha384-wvfXpqpZZVQGK6TAh"
13
14 </head>
15 <body>
16   <app-root></app-root>
17
18
19   <script type="text/javascript" src="assets/js/bootstrap.js"></script>
20 </body>
21 </html>
22

```

All set up let's design the login page UI

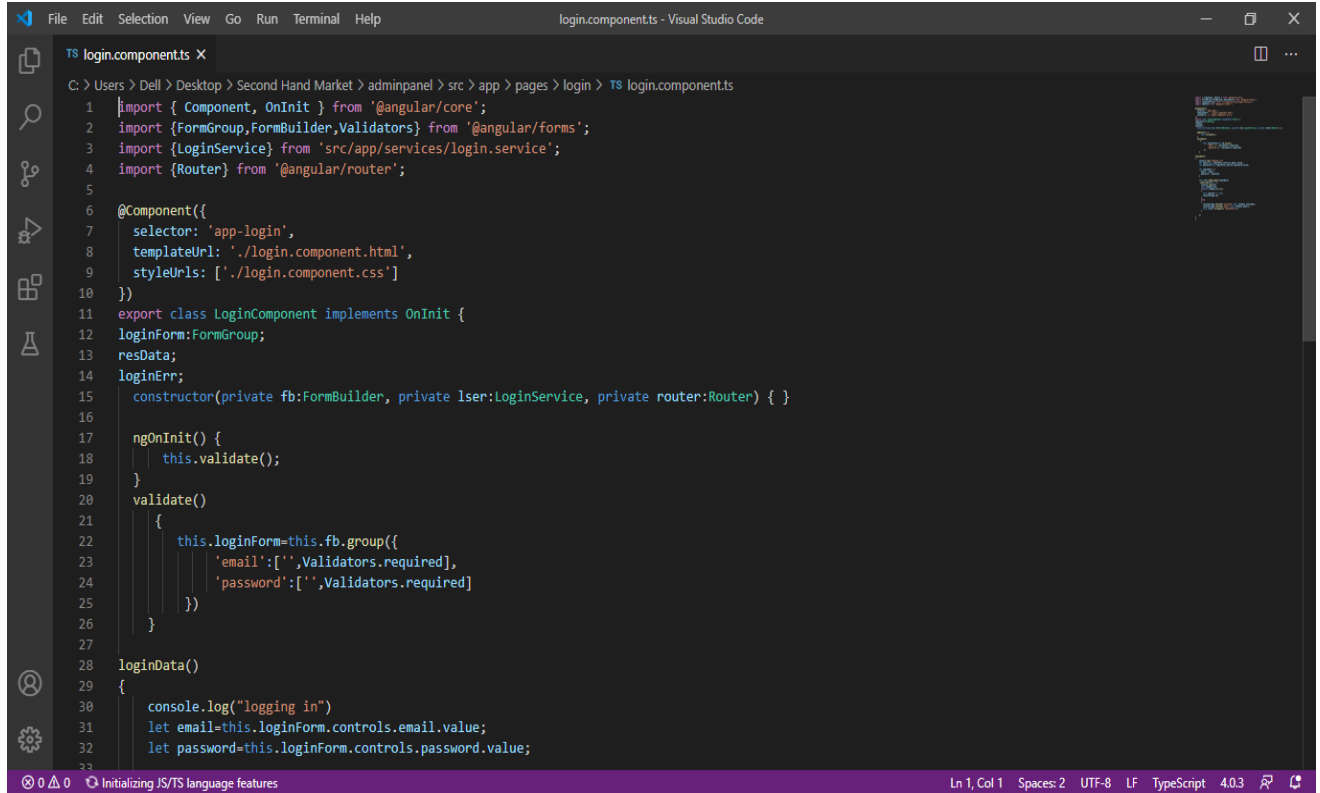
The code below shows the UI design for login page:

```

1  <div class="container">
2    <br>
3    <br>
4    <div class="col-xl-4 col-md-6 col-sm-7 col-12 col-lg-5 Center">
5
6      <h1 class="text-center">Admin Login</h1>
7      <br>
8      <form [formGroup]="loginForm" (ngSubmit)="loginData()">
9        <div class="form-group">
10         <label>Email</label>
11         <input type="text" class="form-control" placeholder="Email" formControlName="email">
12         <div *ngIf="loginForm.controls.email.invalid && loginForm.controls.email.touched">
13           <div *ngIf="loginForm.controls.email.errors.required" class="alert alert-danger">
14             *This field is required
15           </div>
16         </div>
17       </div>
18       <div class="form-group">
19         <label>Password</label>
20         <input type="password" class="form-control" placeholder="password" class="form-control" formControlName="password">
21         <div *ngIf="loginForm.controls.password.invalid && loginForm.controls.password.touched">
22           <div *ngIf="loginForm.controls.password.errors.required" class="alert alert-danger">
23             *This field is required</div>
24         </div>
25       </div>
26       <div *ngIf="loginErr" class="alert alert-danger">
27         Email or password is incorrect
28       </div>
29       <button type="submit" class="btn btn-primary" [disabled]="loginForm.invalid">Login</button>
30       <a href="#" style="margin-left: 37px">Forgot password</a>
31     </form>
32   </div>
33 </div>

```

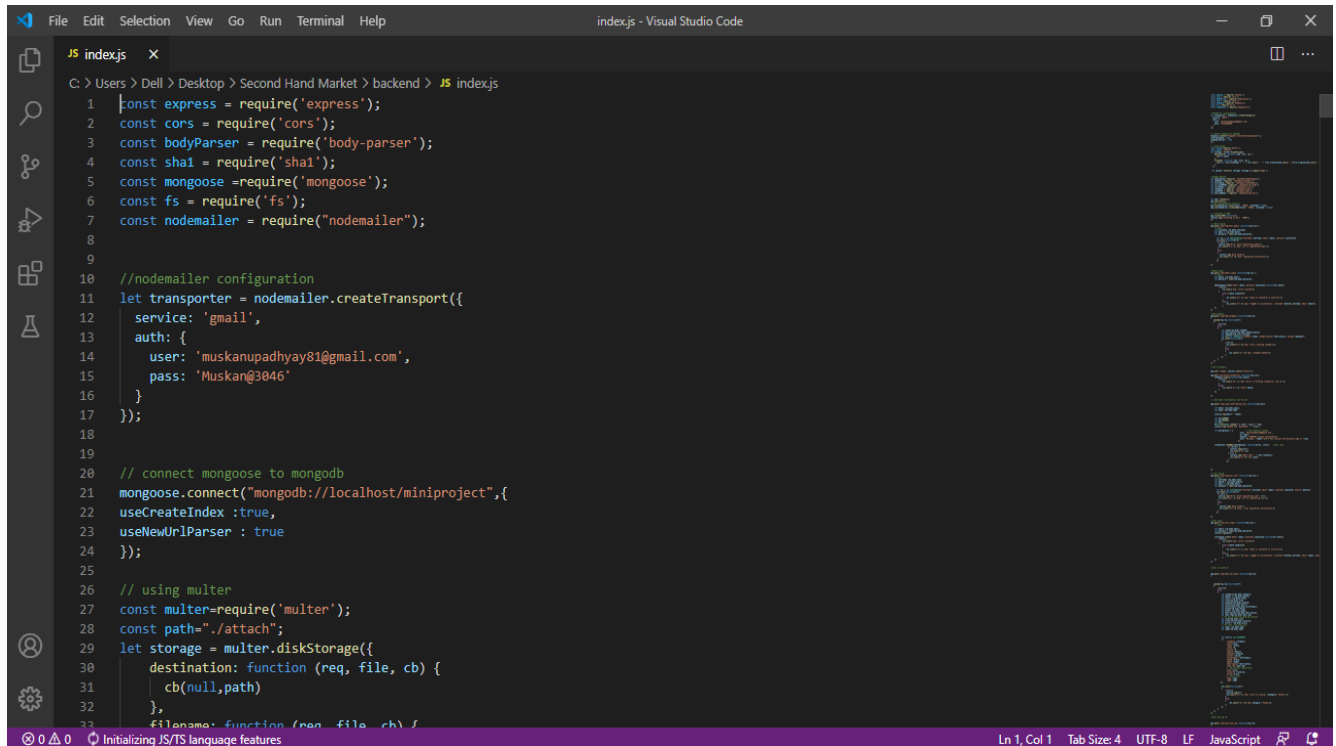
Login.component.ts



```
login.component.ts X
C:\Users> Dell > Desktop > Second Hand Market > adminpanel > src > app > pages > login > TS login.component.ts
1 import { Component, OnInit } from '@angular/core';
2 import { FormGroup, FormBuilder, Validators } from '@angular/forms';
3 import { LoginService } from 'src/app/services/login.service';
4 import { Router } from '@angular/router';
5
6 @Component({
7   selector: 'app-login',
8   templateUrl: './login.component.html',
9   styleUrls: ['./login.component.css']
10 })
11 export class LoginComponent implements OnInit {
12   loginForm: FormGroup;
13   resData;
14   loginErr;
15   constructor(private fb: FormBuilder, private lser: LoginService, private router: Router) { }
16
17   ngOnInit() {
18     this.validate();
19   }
20   validate()
21   {
22     this.loginForm=this.fb.group({
23       'email':['',Validators.required],
24       'password':['',Validators.required]
25     })
26   }
27
28   loginData()
29   {
30     console.log("logging in")
31     let email=this.loginForm.controls.email.value;
32     let password=this.loginForm.controls.password.value;
33   }
```

Backend Api file to handle all the database requests

Below is a code snapshot of the backend api file to handle all the database related queries:



```
1  const express = require('express');
2  const cors = require('cors');
3  const bodyParser = require('body-parser');
4  const sha1 = require('sha1');
5  const mongoose = require('mongoose');
6  const fs = require('fs');
7  const nodemailer = require("nodemailer");
8
9
10 //nodemailer configuration
11 let transporter = nodemailer.createTransport({
12   service: 'gmail',
13   auth: {
14     user: 'muskanupadhyay81@gmail.com',
15     pass: 'Muskan@3046'
16   }
17 });
18
19
20 // connect mongoose to mongodb
21 mongoose.connect("mongodb://localhost/miniproject",{
22   useCreateIndex :true,
23   useNewUrlParser : true
24 });
25
26 // using multer
27 const multer=require('multer');
28 const path="./attach";
29 let storage = multer.diskStorage({
30   destination: function (req, file, cb) {
31     cb(null,path)
32   },
33   filename: function (req, file, cb) {
```

Testing

WEB TESTING, or website testing is checking your web application or website for potential bugs before it's made live and is accessible to general public. Web Testing checks for functionality, usability, security, compatibility, performance of the web application or website.

During this stage issues such as that of web application security, the functioning of the site, its access to handicapped as well as regular users and its ability to handle traffic is checked.

1. **Login:** While log in in to website user need to enter email and password

if(email is incorrect || password is incorrect)

 Generate err window("Email or password is incorrect ");

else

 Login to website

2. **Sign up:** While registering on the website user needs to enter username, email, password, confirm password and mobile no.

if(all details are entered correctly):

 "sends verification code to entered g-mail ID and generate a window to enter the sent verification code"

 if(verification code matches):

 "registered successfully, automatically logged in to website"

else:

Registration fails, user can generate another OTP or change email.

else:

“produce err message to fill all the details correctly.”

3. Post ads: User need to enter all the details given in the form.

if(all the details filled and at least two images of the product are selected):

“Post ad successfully”

else:

“ Ad can’t be posted until the user does not fills the correct details”

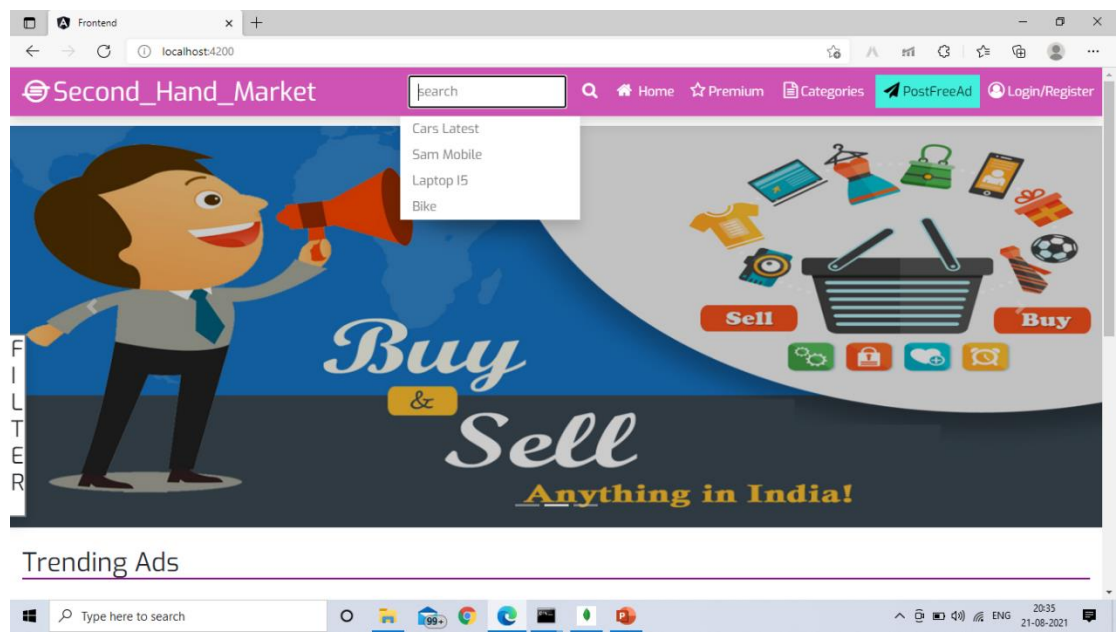
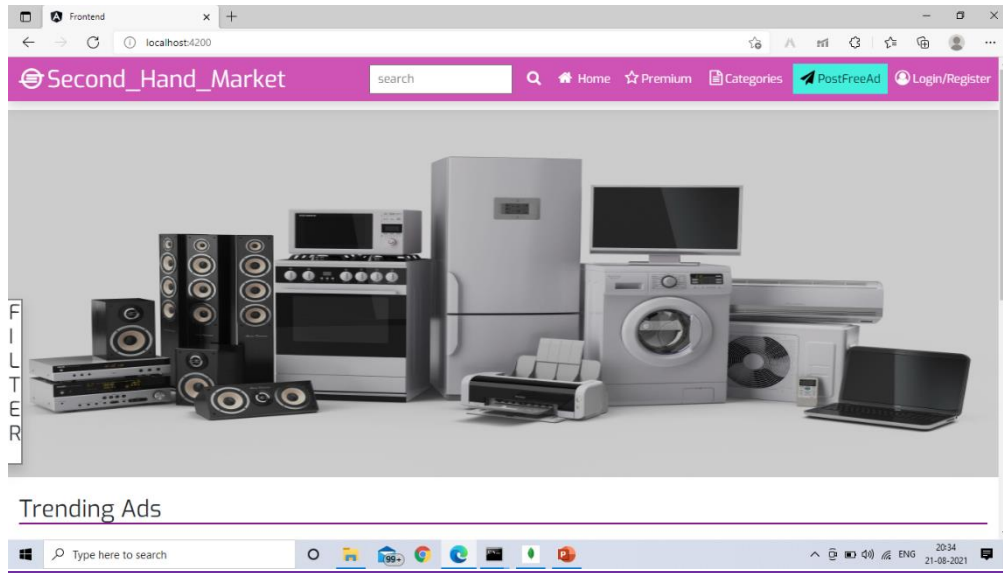
4. chat with other Buyers/Sellers:

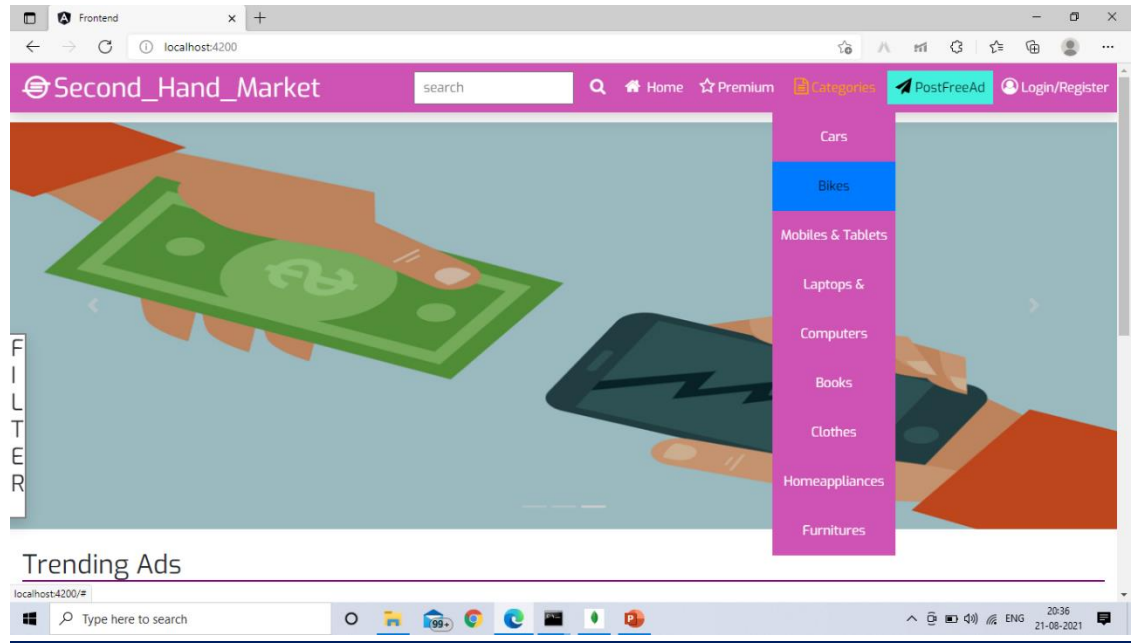
User can send messages to those users only which has some advt. Products on website otherwise the user can’t chat with the seller unnecessarily.

5. Change Password: user need to enter old password, new password and confirm new password if old password matches, new password and confirm new password field has the same password value, password is changed otherwise it generate a error message that old password did not match or new password and confirm password should be same depending on the condition.

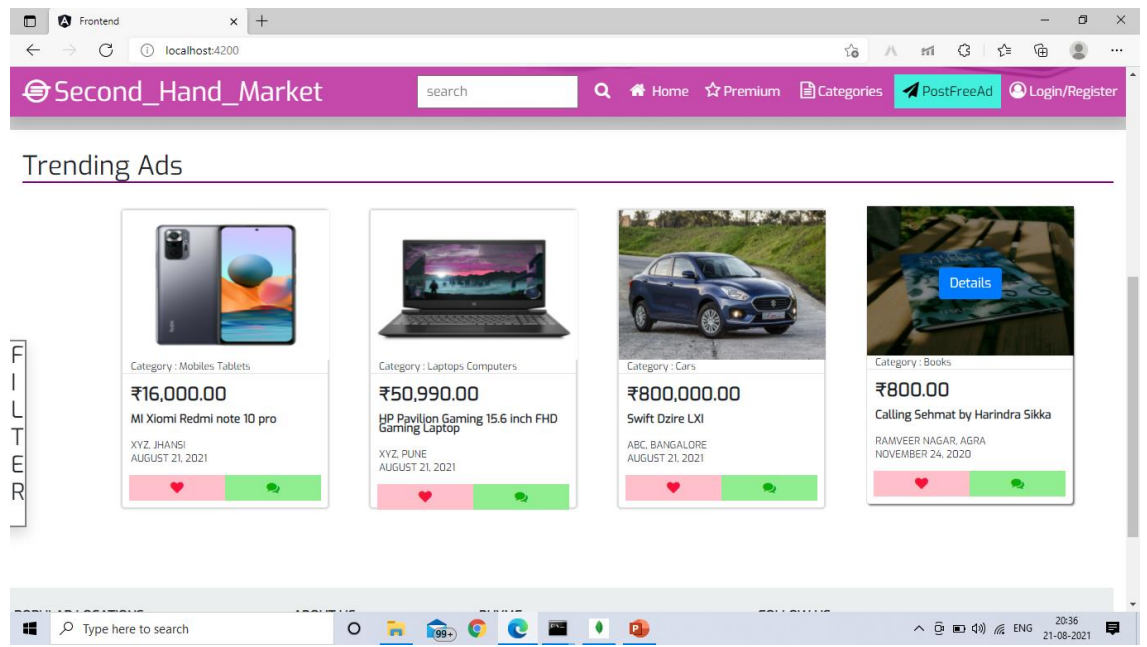
Snapshots:

HOME PAGE or LANDING PAGE

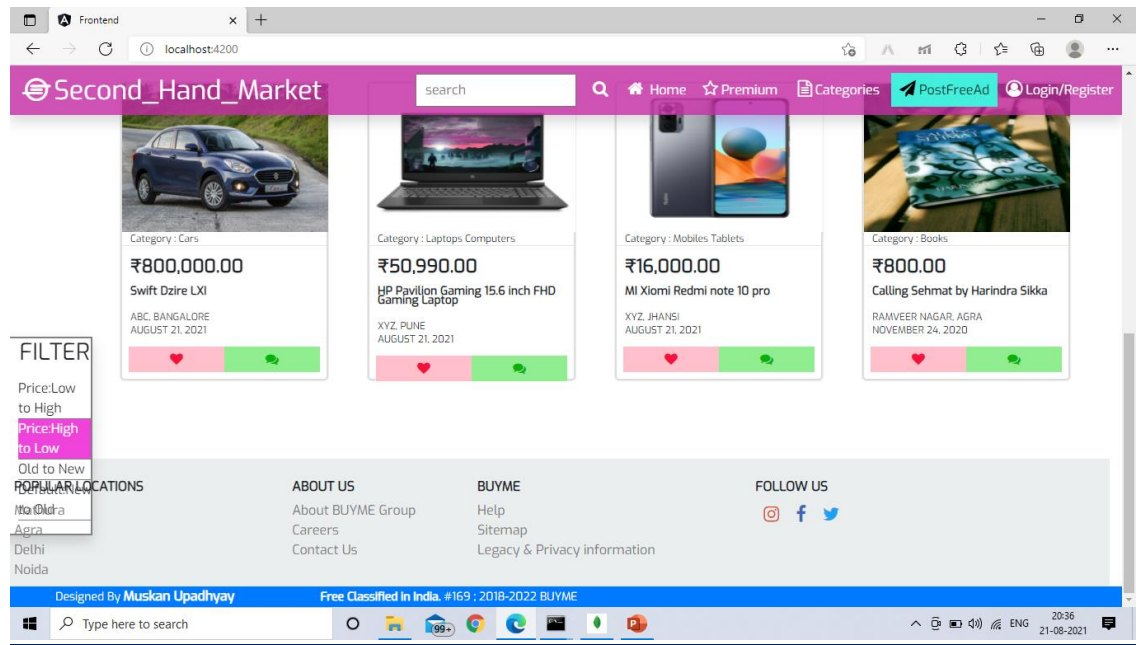




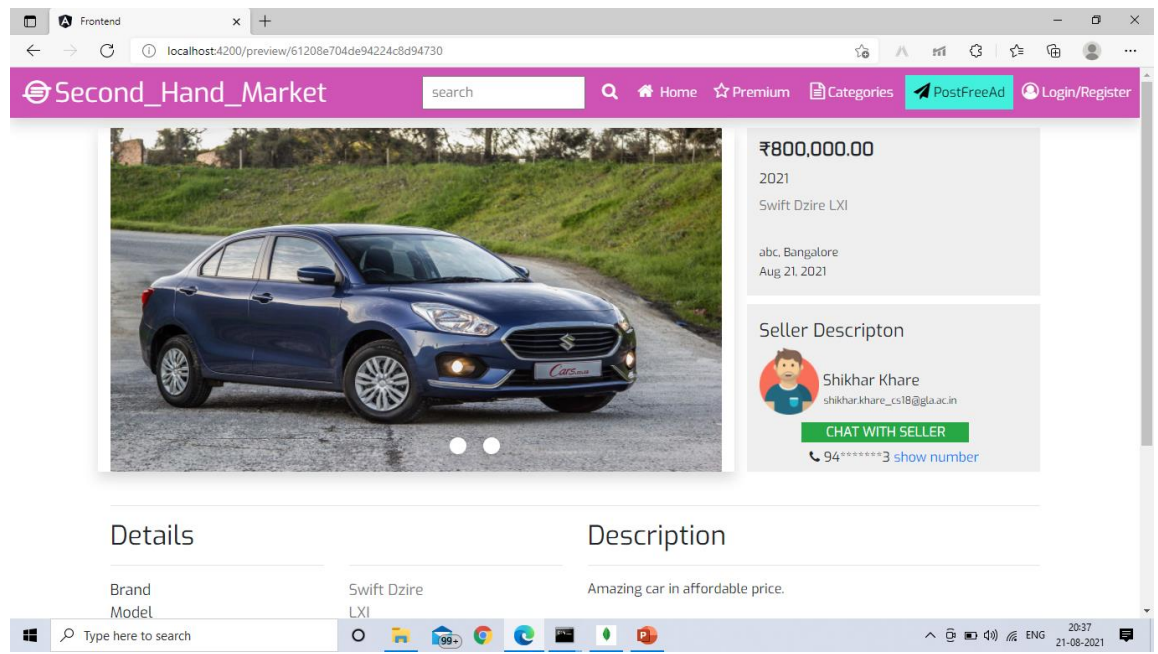
TRENDING ADS



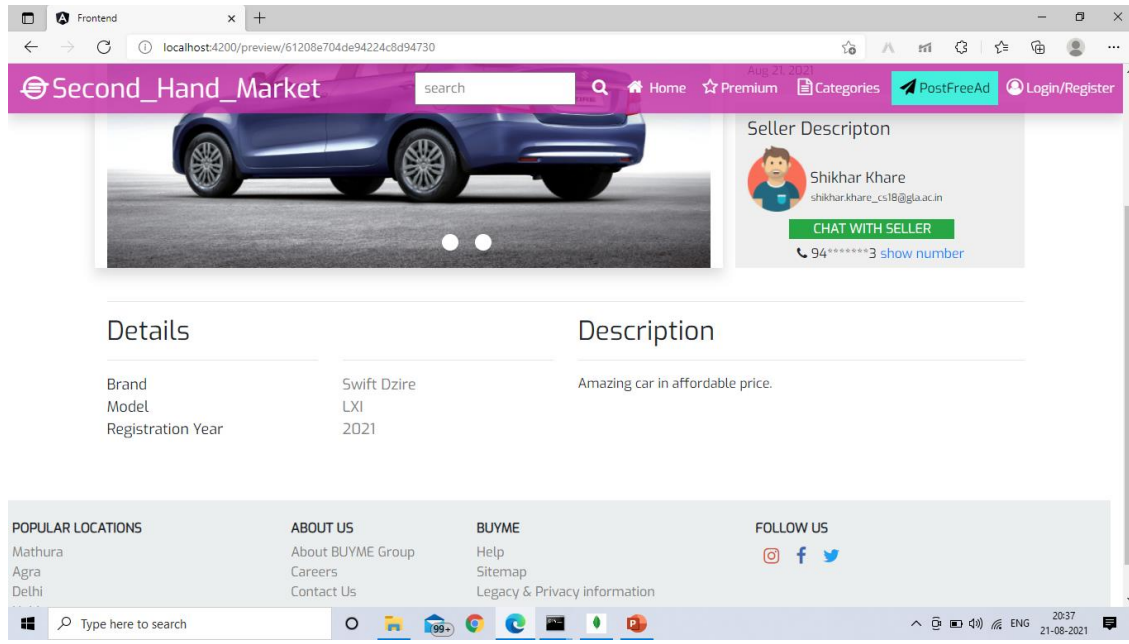
FOOTER (HOME PAGE)



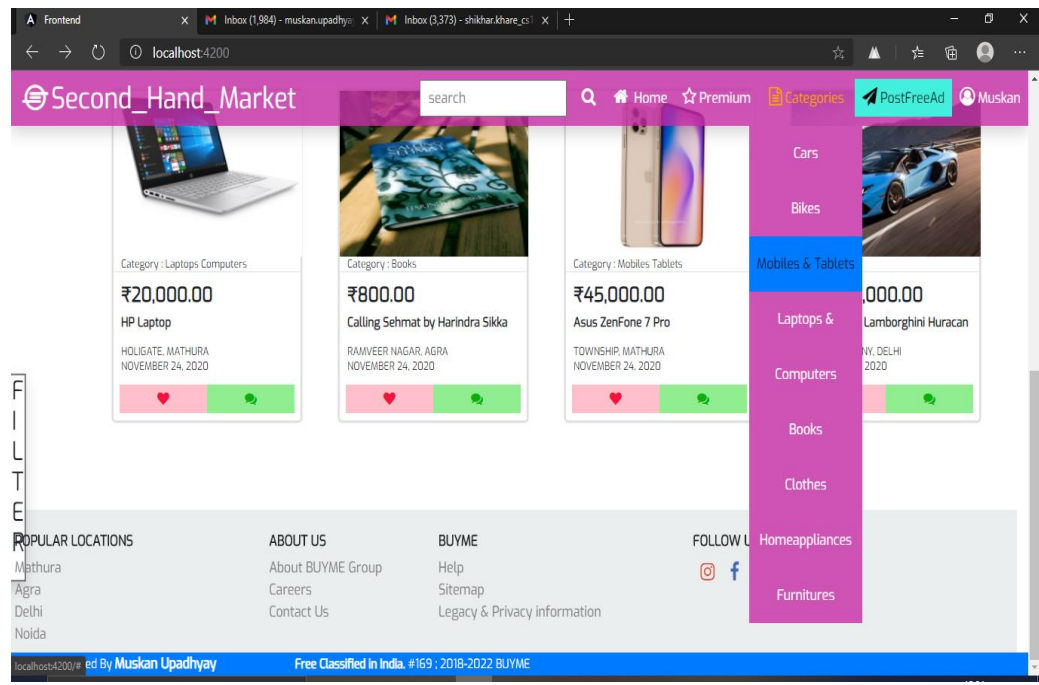
DETAILS OF THE PRODUCT



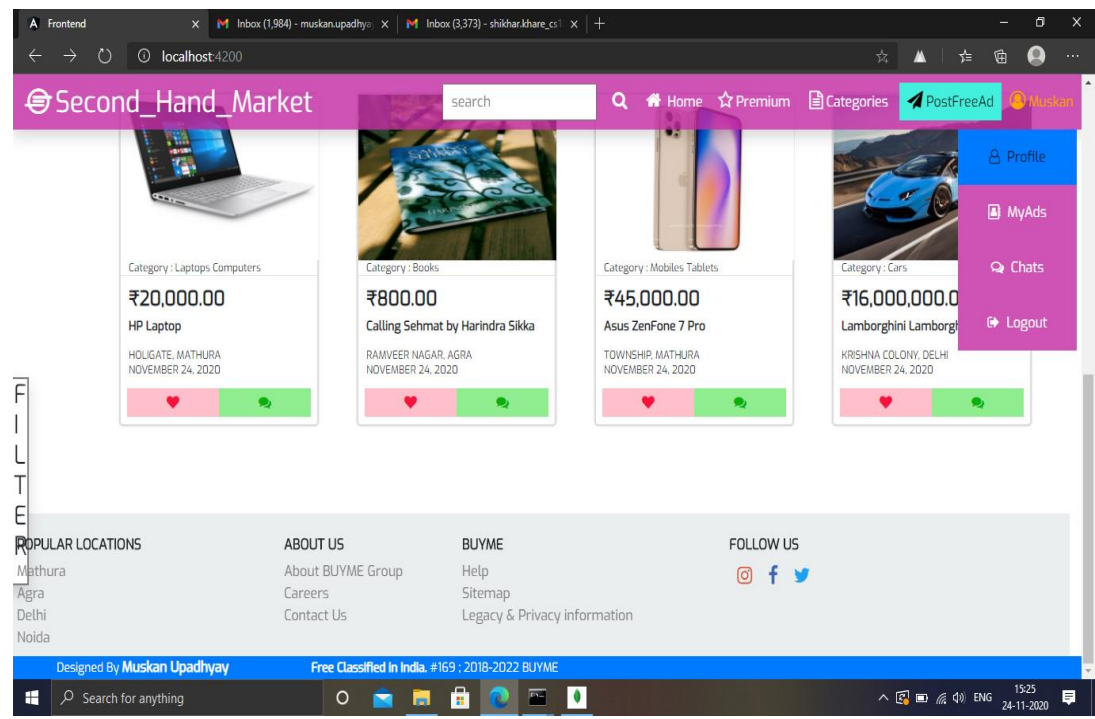
DETAILS, DESCRIPTION, PRICE AND MODEL OF THE PRODUCT



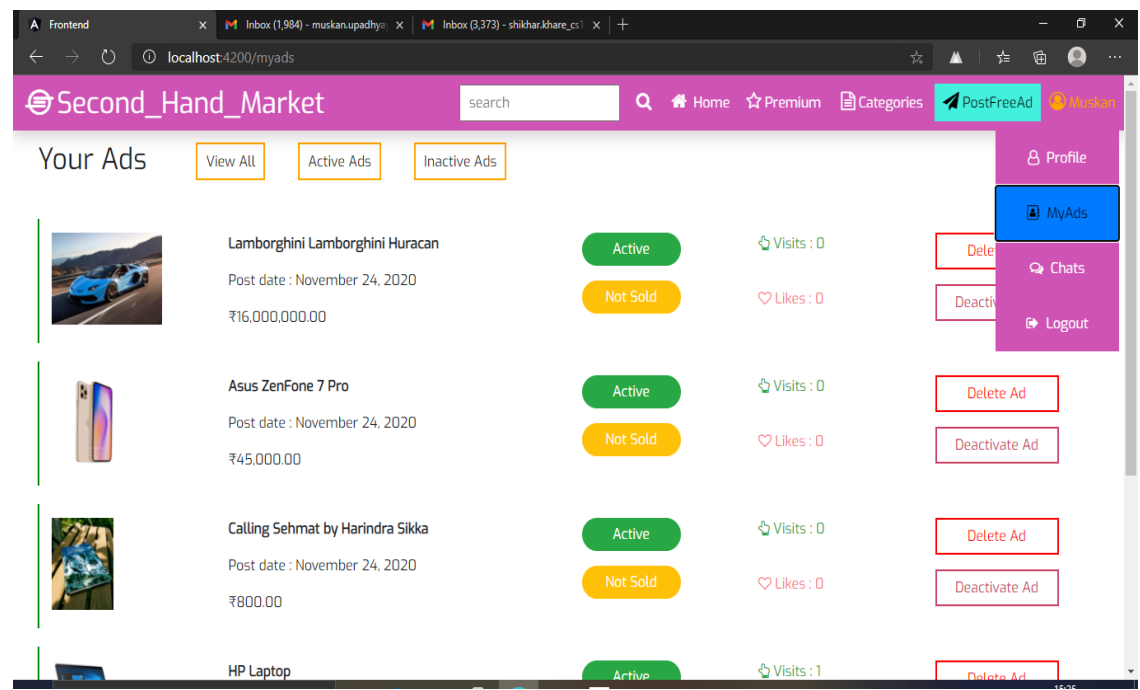
CATEGORIES



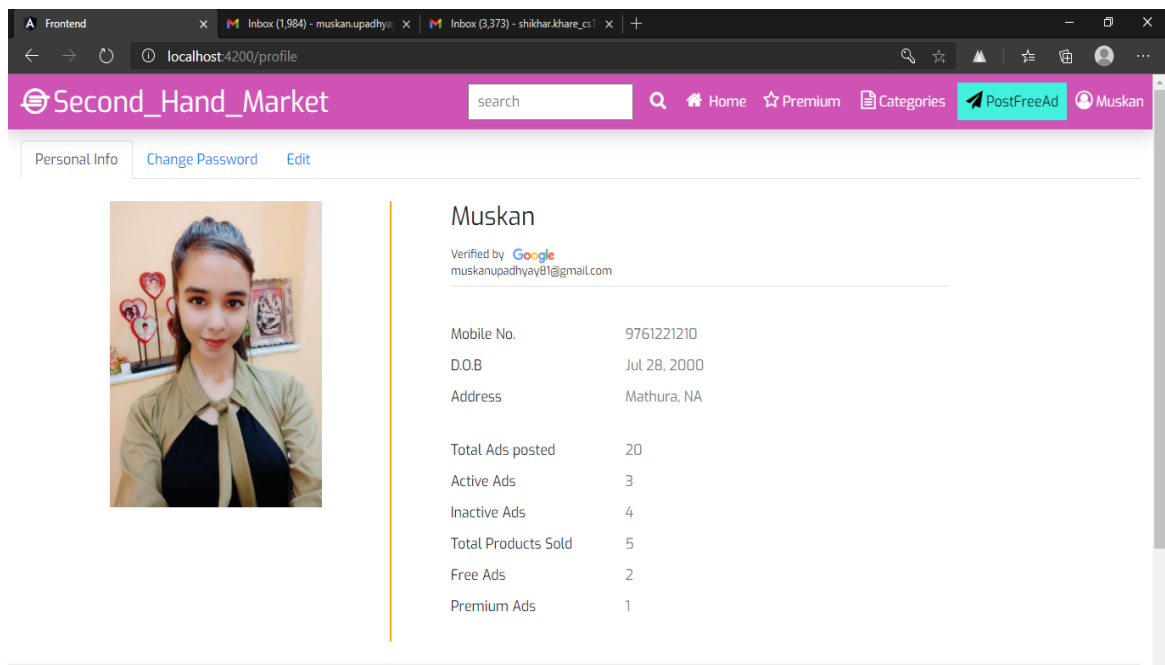
PROFILE



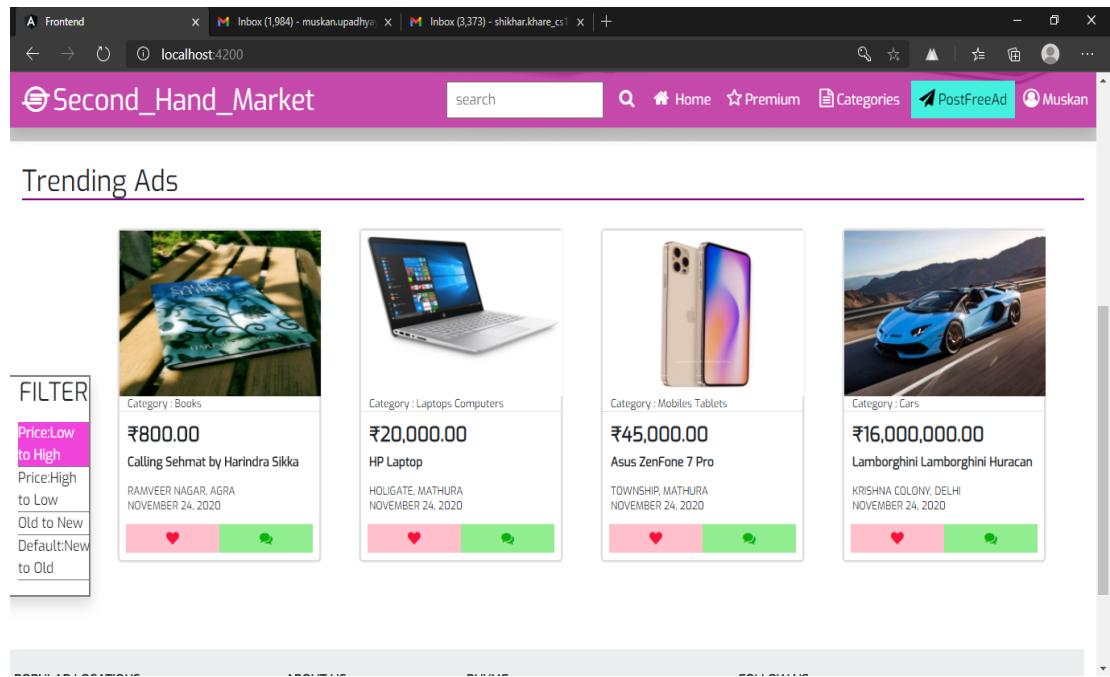
MY ADS



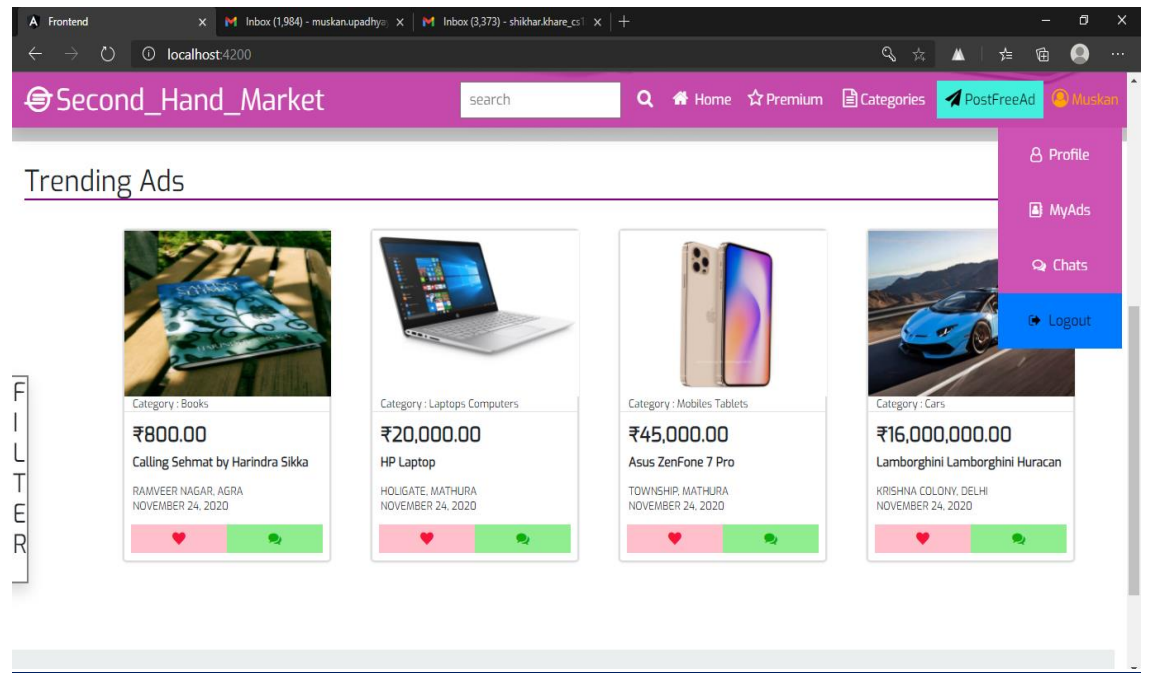
USER'S PROFILE



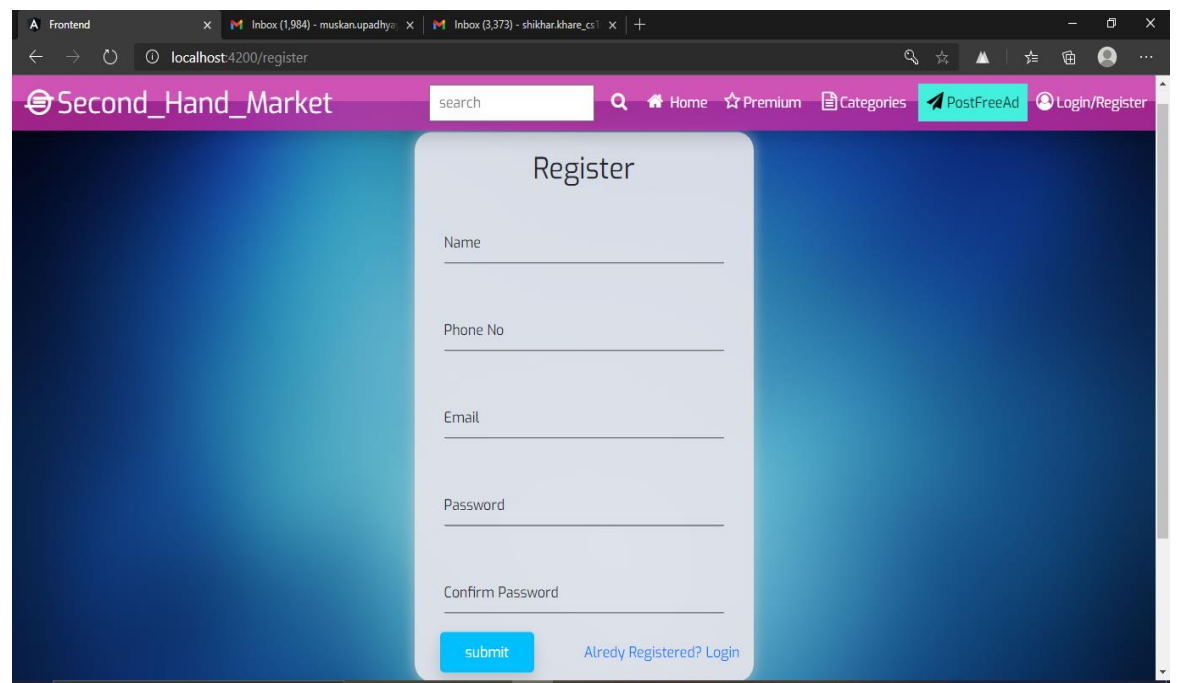
FILTER ADS



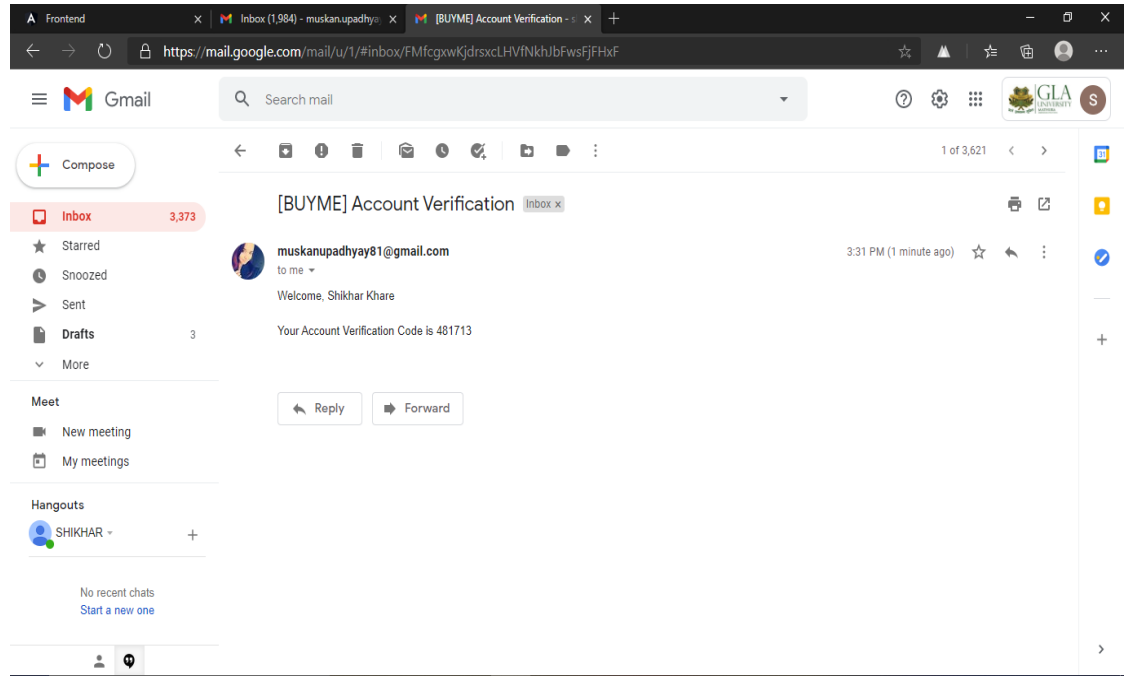
LOGOUT



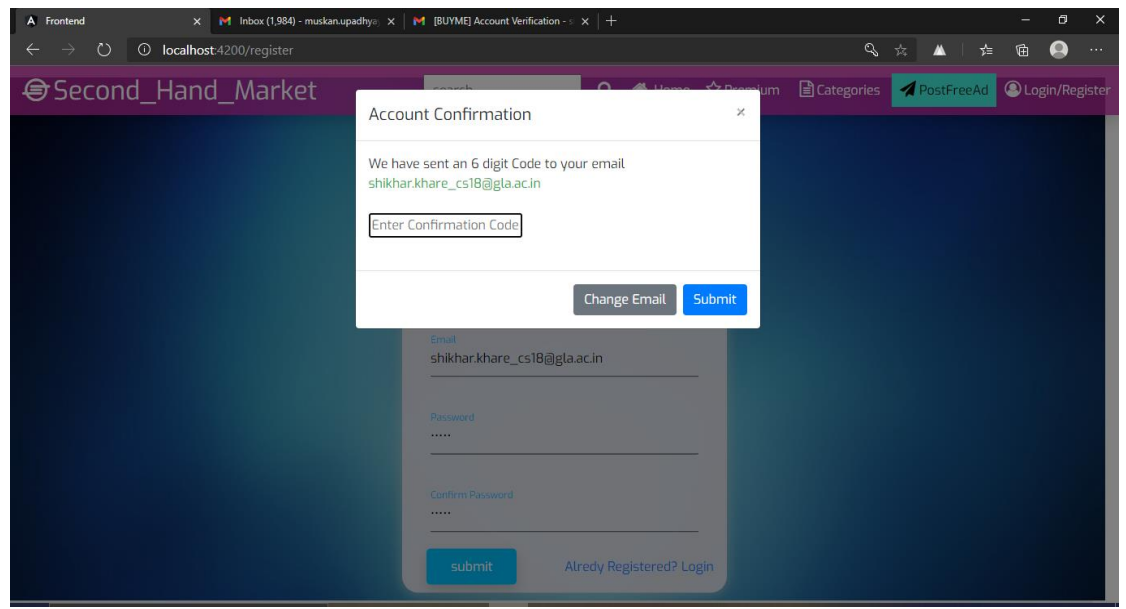
REGISTER

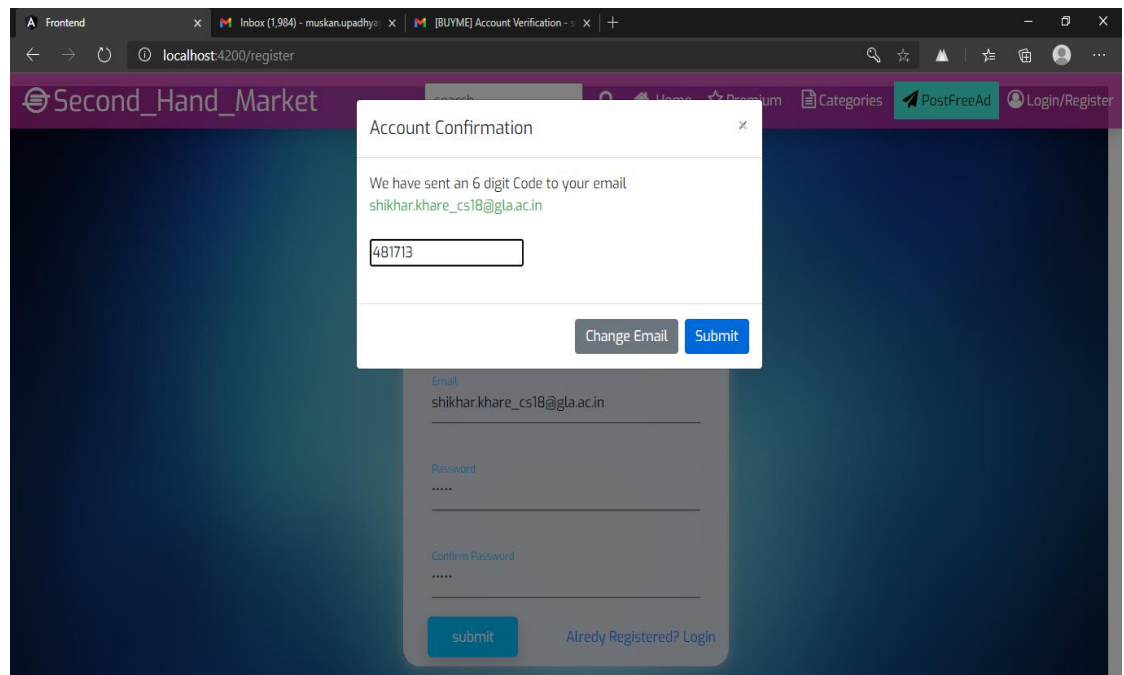


ACCOUNT VERIFICATION ON MAIL

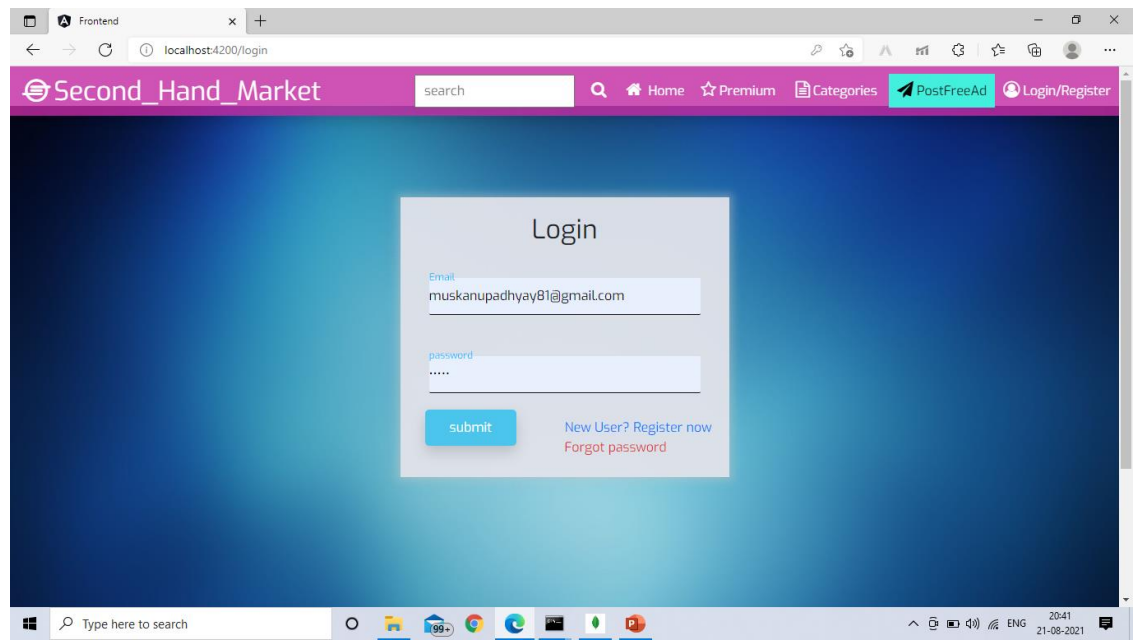


CONFIRMATION CODE





LOGIN PROFILE



EDIT PROFILE

Frontend localhost:4200/profile

Second_Hand_Market search Home Premium Categories PostFreeAd Muskan

Personal Info Change Password Edit

Name
Muskan

Email
muskanupadhyay81@gmail.com

Date of Birth
7/28/2000

Phone
9761221210

Address
Mathura
NA

Change photo

Update Details

CHANGE PASSWORD

Frontend localhost:4200/profile

Second_Hand_Market search Home Premium Categories PostFreeAd Muskan

Personal Info Change Password Edit

Old Password
12345

New Password
.....

Confirm Password
.....

Update Password

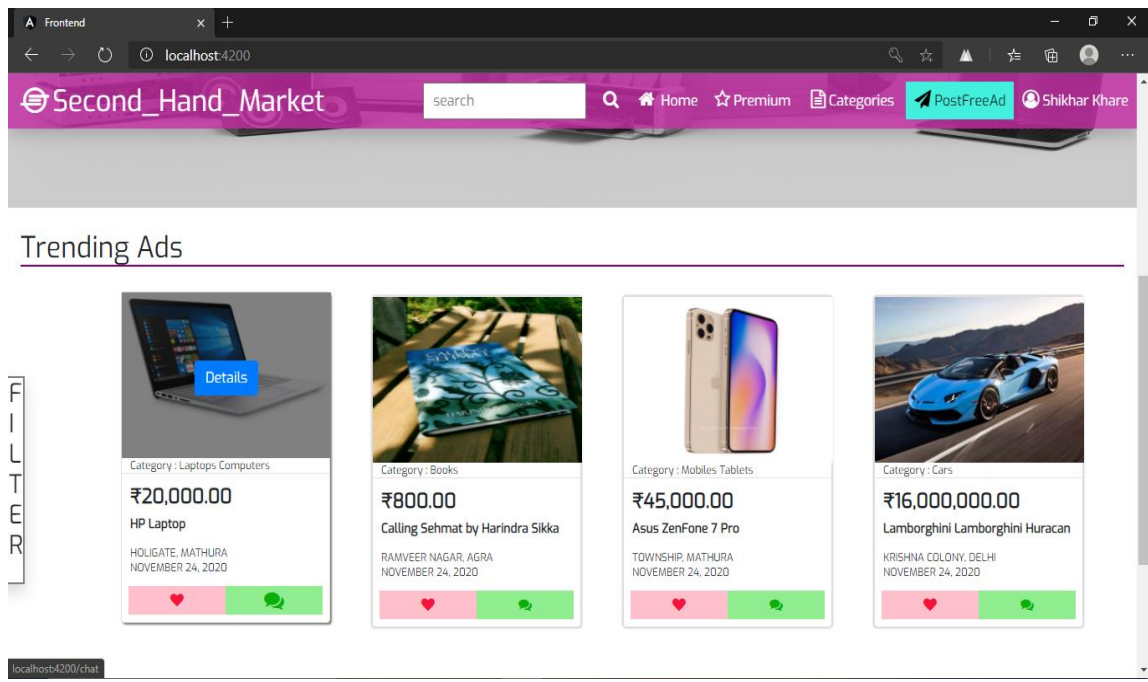
POPULAR LOCATIONS
Mathura
Agra
Delhi
Noida

ABOUT US
About BUYME Group
Careers
Contact Us

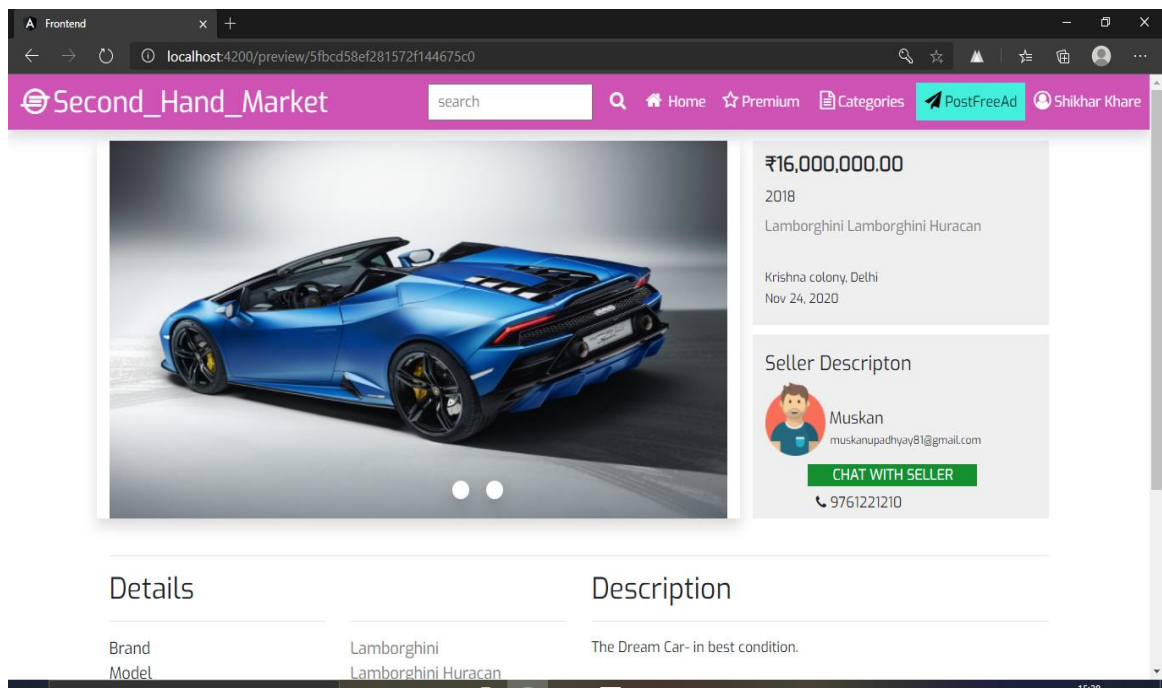
BUYME
Help
Sitemap
Legacy & Privacy information

FOLLOW US
Instagram Facebook Twitter

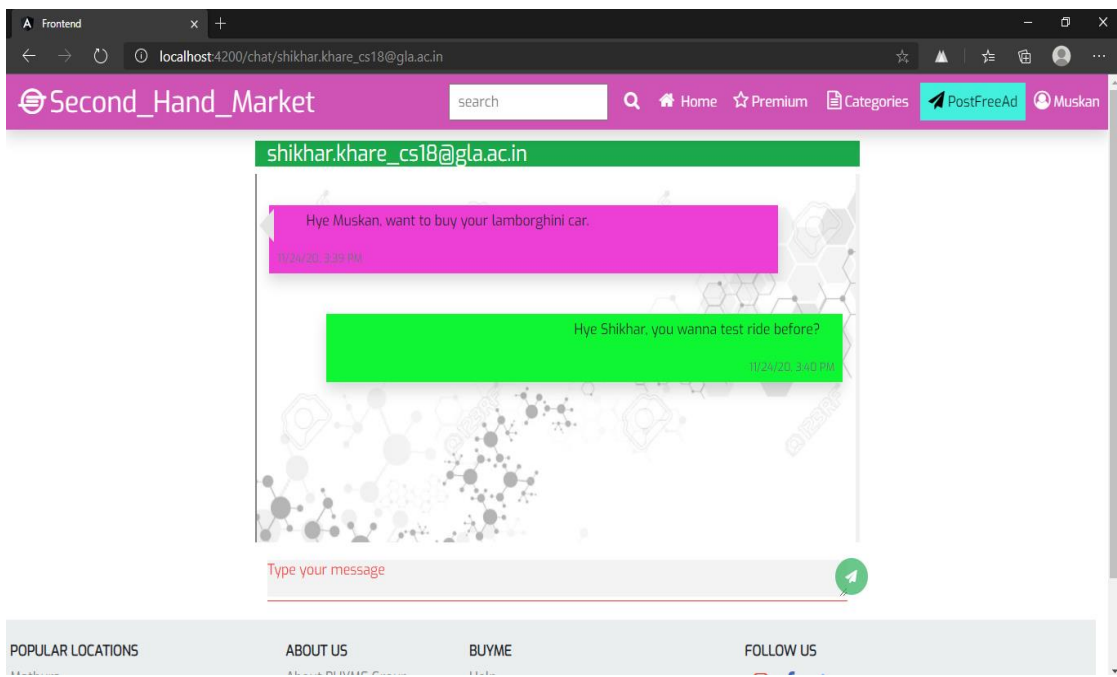
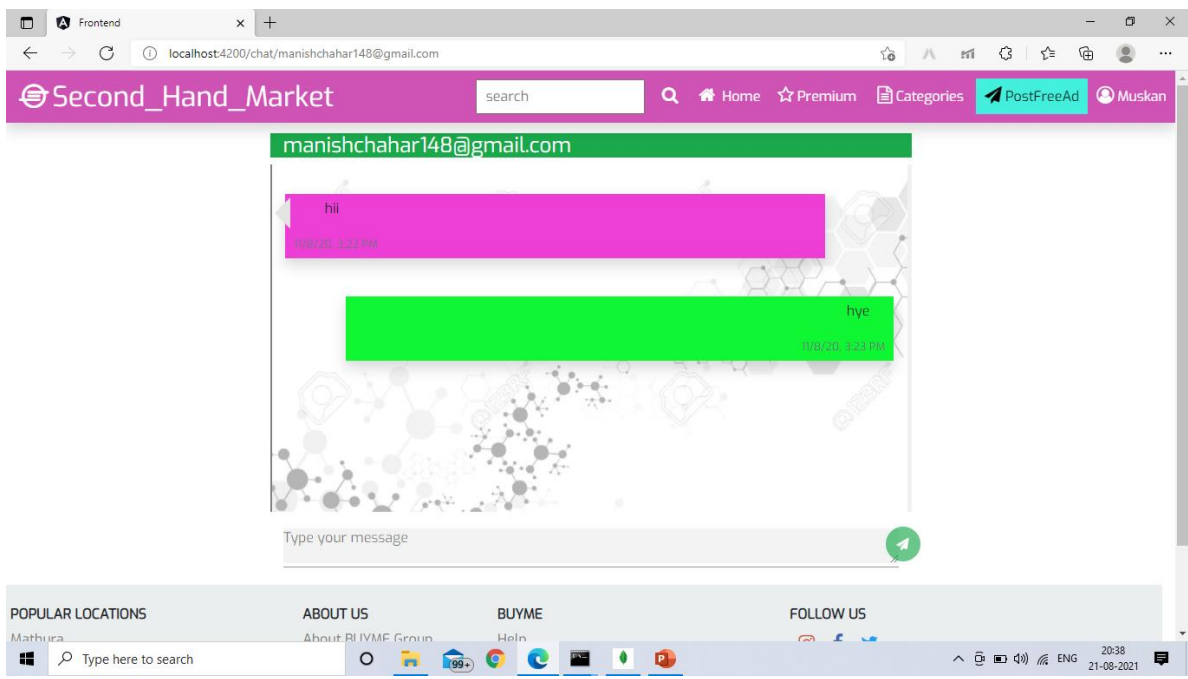
LIKE BUTTON



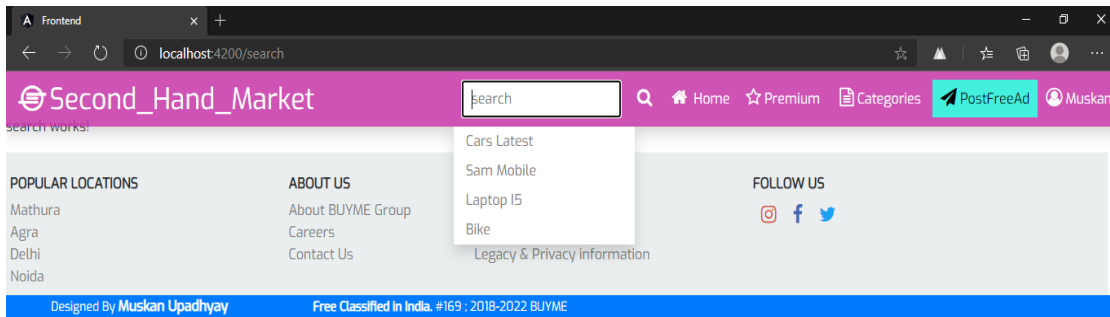
DETAILS OF POST AD



CHAT WITH SELLER



SEARCH YOUR PRODUCT



Difficulties Encountered:

- We faced problem in preserving the login state. When a user input the data in login component, if the user refreshes the website, then it will be logged out automatically. We tackled this problem by using local storage. After refreshing, the data gets stored in local storage so we can read it from there and the previous data will not get erased.
- Another difficulty we faced is in Authentication (passing token during login). Due to less time, we are not able to implement it but we can do it by using JWT or JSON Web Token (a token used for authentication).

Contributions

Our group member MUSKAN UPADHYAY had contributed in both frontend and backend development. Backend development of the project has been done by her. She has also made the code for chat section in which a person can send chat with the seller of the product if he wants to buy it. We used Node.js for backend. She made the logic and the algorithms very sharply which are working properly. Muskan has done a great work in this backend section. Also, she made significant contributions in the designing of frontend of the website including the profile update feature, chat UI, and products display card etc and the most important part the responsiveness of the website is also taken into account which makes our website work properly on any kind of display without any problem. She has applied proper data formats and different types of validations to prevent our website from any kind of exception and invalid data inputs. She has also implemented feature of user verification with gmail using the verification code which is sent to user's gmail ID when registering on our website. She has designed filters and search methods so that user can find the required products easily without unwanted scrolls on our site thus enhancing user experience. She had made her full efforts to make the website more user friendly and interactive to enhance user experience.

Our group member NITIN KUMAR SINGH had contributed in this project in the implementation of both frontend and backend. whole admin panel front end has been made by him by html, Bootstrap and CSS. The backend related part of creation of schema of sellers and insertion of products uploaded by a particular seller in that schema has been done by him. He has made the validation code to check whether the person logging in website is present or not, if a person is not present the

message of signing up will be displayed otherwise that person get logged in and can see his/her profile. This task is related to sending the entered information on backend and then check for the presence of entered data in particular collections using query and then send the response as per the response of query. While implementing frontend profile of user he designed functionalities for users to update his required details as per his wish. He implemented this task using update query by which the data stored in data base can be changed.

FUTURE SCOPE

- **Buying and Selling of Products:** Buying and selling of products module is used to maintain member registration, product search, listing of products to sell, buy product, display order status.
- **Back Office and Administration Module:** This module will be used by the back-office user and system administrator to approve the product, member, update courier details, payment processing.

References

This project is the output of work of our team members. But there're some websites which helped us a lot.

These are:

1. www.angular.io
2. www.w3schools.com
3. www.nodejs.org
4. www.monogodb.com
5. www.youtube.com
6. www.material.angular.com
7. www.npmjs.com
8. www.bootstrap.com
9. www.stackoverflow.com
10. www.googlefonts.com